

# Máster Avanzado de Programación en Python para Hacking, BigData y Machine Learning

FUNDAMENTOS DE PYTHON

## LECCIÓN 02

# POSICIÓN ESTRATÉGICA DE PYTHON. PREPARANDO EL ENTORNO DE DESARROLLO.

# ÍNDICE

Introducción

Objetivos

Posición estratégica del lenguaje Python

Preparando el entorno de desarrollo

Instalar Python en Ubuntu

Instalar Python en Windows

Conclusiones

# INTRODUCCIÓN

En esta lección haremos un breve repaso sobre la posición estratégica del lenguaje Python, destacando las áreas donde es aplicado.

Por otro lado, prepararemos el entorno de desarrollo en nuestras máquinas, para poder instalar tanto el lenguaje Python como su intérprete, necesario para poder ejecutar su código. Para facilitar la tarea del alumnado, se mostrará una guía de instalación de Python para los sistemas operativos: Windows 10 y Ubuntu 16.04.

## OBJETIVOS

Al finalizar esta lección serás capaz de:

- 1 Conocer la posición estratégica de Python.
- 2 Conocer sus principales áreas de aplicación.
- 3 Saber instalar Python en cualquier sistema operativo.
- 4 Gestionar los paquetes de Python.

## Posición estratégica del lenguaje Python

- El lenguaje de programación Python ha logrado ser el centro de atención de muchos desarrolladores de software, debido a su alta interpretabilidad y facilidad de uso.
- Puesto que Python es código abierto, se ha podido liberar una gran cantidad de módulos y librerías diseñados para resolver problemas de diferente índole.
- Existen varios mecanismos para medir la popularidad de los lenguajes de programación, como el índice TIOBE o PYPL.

## Posición estratégica del lenguaje Python

PYPL mide la popularidad de los lenguajes de programación teniendo en cuenta con qué frecuencia los desarrolladores de software buscan tutoriales en Google sobre un lenguaje de programación.

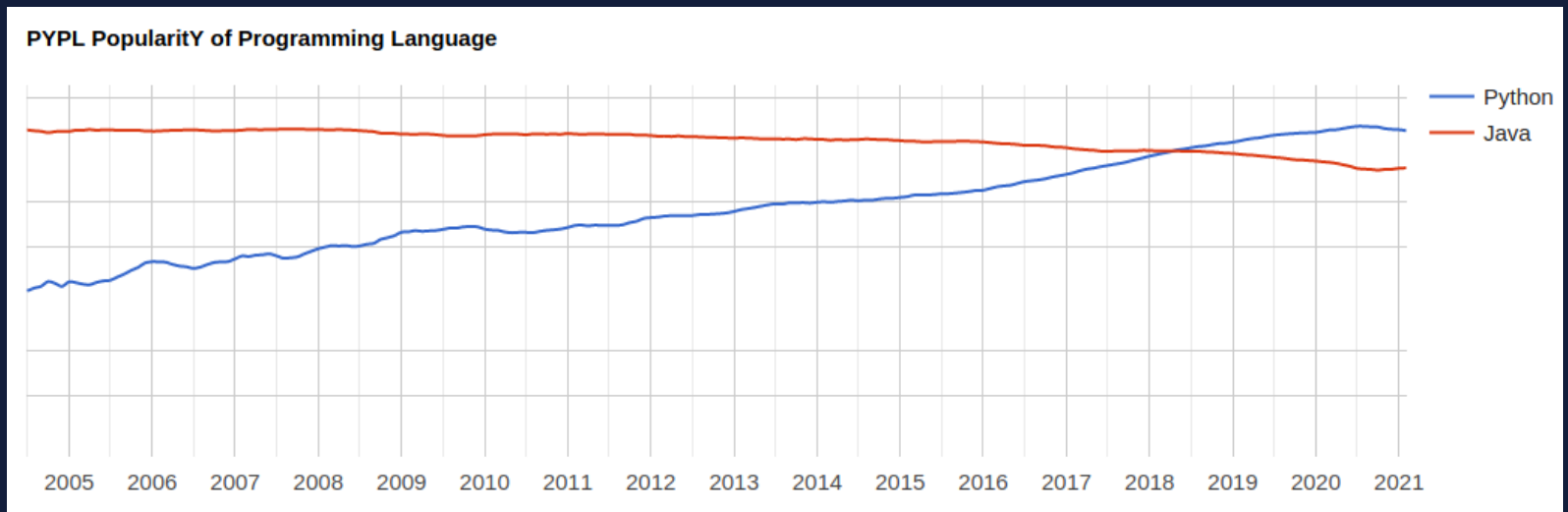
Worldwide, Feb 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	30.06 %	+0.3 %
2		Java	16.88 %	-1.7 %
3		JavaScript	8.43 %	+0.4 %
4		C#	6.69 %	-0.6 %
5	↑	C/C++	6.5 %	+0.5 %

<https://pypl.github.io/PYPL.html>

## Posición estratégica del lenguaje Python

Python no siempre ha sido considerado como el lenguaje de programación más utilizado o famoso, si no que ha necesitado una larga trayectoria para ganarse este puesto.





## Posición estratégica del lenguaje Python

El índice TIOBE calcula la popularidad de un lenguaje de programación atendiendo a la cantidad de sitios indexados en Google, Yahoo y Bing que mencionan un lenguaje.

Feb 2021	Feb 2020	Change	Programming Language	Ratings	Change
1	2	▲	C	16.34%	-0.43%
2	1	▼	Java	11.29%	-6.07%
3	3		Python	10.86%	+1.52%
4	4		C++	6.88%	+0.71%
5	5		C#	4.44%	-1.48%
6	6		Visual Basic	4.33%	-1.53%

<https://www.tiobe.com/tiobe-index/>

## Posición estratégica del lenguaje Python

El hecho de que Python haya ido ganando adeptos en los últimos años puede deberse, entre otros, a su alta versatilidad, ya que puede ser utilizado en diferentes disciplinas.

Entre las disciplinas donde podemos utilizar Python destacamos:

- Machine Learning
- Big Data
- Hacking

## Posición estratégica del lenguaje Python

En relación a la especialidad de Machine Learning, podemos encontrar una gran variedad de librerías que permiten implementar de una forma sencilla técnicas avanzadas como redes neuronales, regresiones o técnicas de clustering, entre otros.

Algunas de las librerías de Python que pueden resultar de interés para resolver problemas de Machine Learning y visualización son las siguientes:

- Scikit-learn
- Matplotlib
- TensorFlow
- Etc.

## Posición estratégica del lenguaje Python

A lo largo de este máster estudiaréis en detalle sobre cómo utilizar Python para resolver problemas relacionados con hacking, a continuación mostraremos algunas de las herramientas más conocidas que son utilizadas en Python para este propósito:

- SCAPY
- IMPACKET
- Requests
- Cryptography

## **Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04**

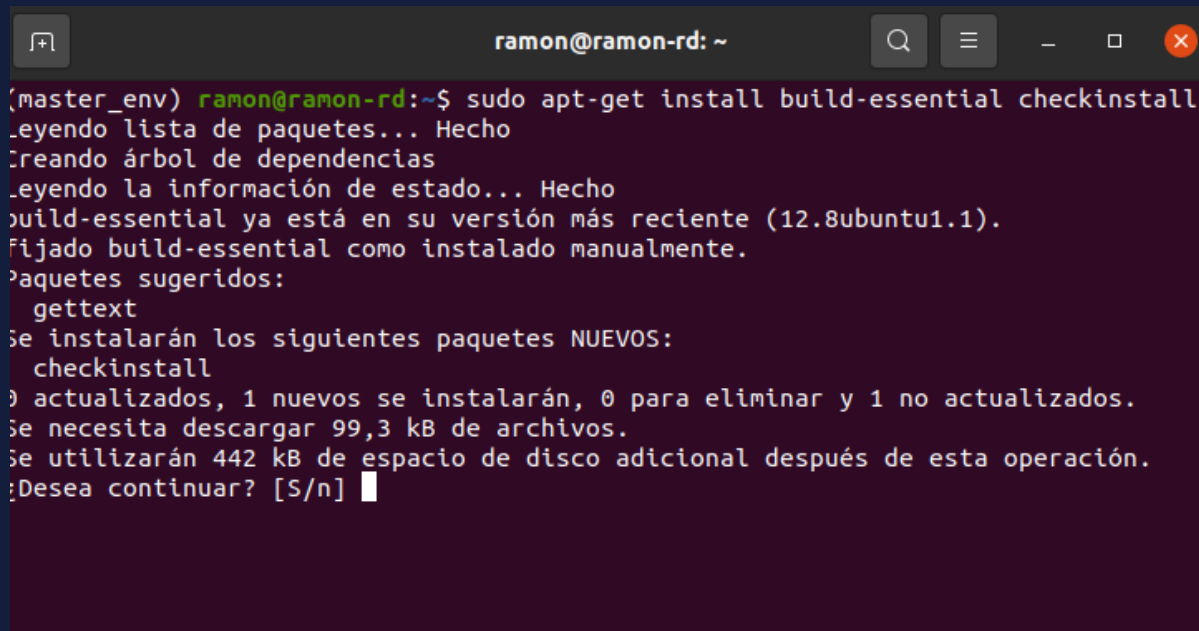
Debido a la gran importancia e influencia que ha tenido Python en programación, la mayoría de las distribuciones de Ubuntu incluyen una versión preinstalada, aunque suele ser antigua.

Para conocer la versión instalada por defecto ejecutaremos en la terminal el comando: *python -V*

En algunos sistemas podréis encontrar instalada la versión 2 o 3.5 de Python. En estos casos, debéis seguir este manual para instalar la versión 3.8.7. Si viniera instalada por defecto, podéis pasar al siguiente apartado.

## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 1: actualizamos los paquetes con la orden “sudo apt-get update” y posteriormente instalamos los paquetes “build-essential” y “checkinstall”



```
ramon@ramon-rd: ~  
(master_env) ramon@ramon-rd:~$ sudo apt-get install build-essential checkinstall  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
build-essential ya está en su versión más reciente (12.8ubuntu1.1).  
Fijado build-essential como instalado manualmente.  
Paquetes sugeridos:  
  gettext  
Se instalarán los siguientes paquetes NUEVOS:  
  checkinstall  
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 1 no actualizados.  
Se necesita descargar 99,3 kB de archivos.  
Se utilizarán 442 kB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n]
```

# Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 2: Instalamos los paquetes restantes que son necesarios para que Python pueda funcionar:

```
ramon@ramon-rd: ~  
(master_env) ramon@ramon-rd:~$ sudo apt-get install libreadline-gplv2-dev libncu  
rsw5-dev libssl-dev libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev lib  
ffi-dev zlib1g-dev  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
libc6-dev ya está en su versión más reciente (2.31-0ubuntu9.2).  
Fijado libc6-dev como instalado manualmente.  
Se instalarán los siguientes paquetes adicionales:  
  bzip2-doc libexpat1-dev libfontconfig1-dev libfreetype-dev libfreetype6-dev  
  libice-dev libncurses-dev libpng-dev libpng-tools libpthread-stubs0-dev  
  libreadline5 libsm-dev libtcl8.6 libtk8.6 libx11-dev libxau-dev libxcb1-dev  
  libxdmcp-dev libxext-dev libxft-dev libxrender-dev libxss-dev libxt-dev tcl  
  tcl-dev tcl8.6 tcl8.6-dev tk tk8.6 tk8.6-dev uuid-dev x11proto-core-dev  
  x11proto-dev x11proto-scrnsaver-dev x11proto-xext-dev xorg-sgml-doctools  
  xtrans-dev  
Paquetes sugeridos:  
  freetype2-doc libice-doc ncurses-doc libsm-doc sqlite3-doc libssl-doc  
  libx11-doc libxcb-doc libxext-doc libxt-dev tcl-dev tcl-tclreadline  
  tcl8.6-doc tk-doc tk8.6-doc  
Se instalarán los siguientes paquetes NUEVOS:  
  bzip2-doc libbz2-dev libexpat1-dev libffi-dev libfontconfig1-dev  
  libfreetype-dev libfreetype6-dev libgdbm-dev libice-dev libncurses-dev  
  libncursesw5-dev libpng-dev libpng-tools libpthread-stubs0-dev  
  libreadline-gplv2-dev libreadline5 libsm-dev libsqlite3-dev libssl-dev  
  libtcl8.6 libtk8.6 libx11-dev libxau-dev libxcb1-dev libxdmcp-dev  
  libxext-dev libxft-dev libxrender-dev libxss-dev libxt-dev tcl tcl-dev  
  tcl8.6 tcl8.6-dev tk tk-dev tk8.6 tk8.6-dev uuid-dev x11proto-core-dev  
  x11proto-dev x11proto-scrnsaver-dev x11proto-xext-dev xorg-sgml-doctools  
  xtrans-dev zlib1g-dev  
0 actualizados, 46 nuevos se instalarán, 0 para eliminar y 1 no actualizados.  
Se necesita descargar 10,6 MB de archivos.  
Se utilizarán 44,4 MB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n]
```

## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 3: Descargamos la versión 3.8.7 desde la página oficial de Python. Podemos hacerlo manualmente o mediante comandos en una terminal.



```
ramon@ramon-rd: /opt
(master_env) ramon@ramon-rd:~$ cd /opt
(master_env) ramon@ramon-rd:/opt$ sudo wget https://www.python.org/ftp/python/3.8.7/Python-3.8.7.tgz
--2021-02-11 20:15:14-- https://www.python.org/ftp/python/3.8.7/Python-3.8.7.tgz
Resolviendo www.python.org (www.python.org)... 151.101.132.223, 2a04:4e42:1f::223
Conectando con www.python.org (www.python.org)[151.101.132.223]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 24468684 (23M) [application/octet-stream]
Guardando como: "Python-3.8.7.tgz"

Python-3.8.7.tgz      100%[=====>]  23,33M  8,78MB/s   en 2,7s

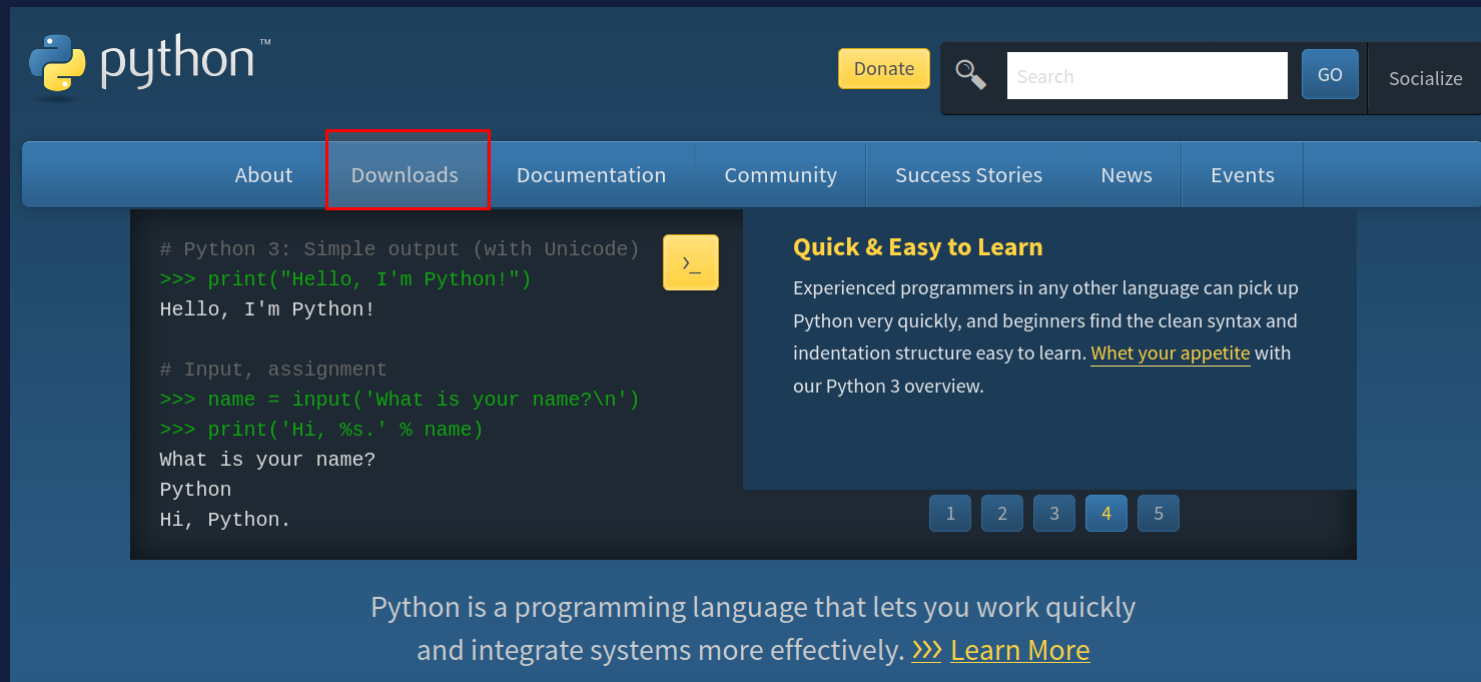
2021-02-11 20:15:17 (8,78 MB/s) - "Python-3.8.7.tgz" guardado [24468684/24468684]

(master_env) ramon@ramon-rd:/opt$
```



## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 3: Descargamos la versión 3.8.7 desde la página oficial de Python.



The screenshot shows the Python.org homepage. The 'Downloads' link in the navigation bar is highlighted with a red box. Below the navigation bar, there is a code editor showing Python 3 code examples and their output. To the right of the code editor, there is a section titled 'Quick & Easy to Learn' with a brief description of Python. At the bottom of the page, there is a statement about Python's purpose and a link to 'Learn More'.

```
# Python 3: Simple output (with Unicode)
>>> print("Hello, I'm Python!")
Hello, I'm Python!

# Input, assignment
>>> name = input('What is your name?\n')
>>> print('Hi, %s.' % name)
What is your name?
Python
Hi, Python.
```

**Quick & Easy to Learn**

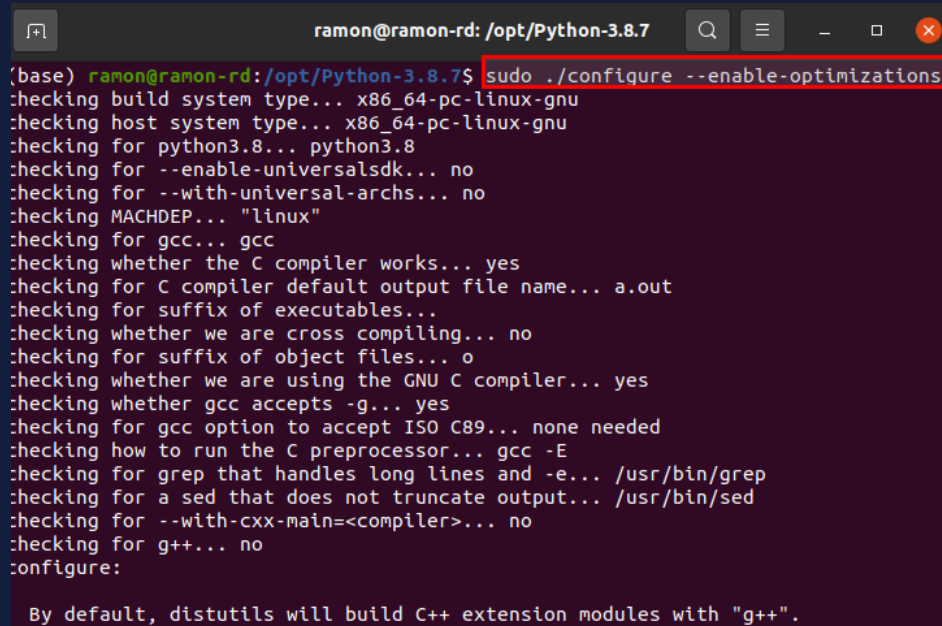
Experienced programmers in any other language can pick up Python very quickly, and beginners find the clean syntax and indentation structure easy to learn. [Whet your appetite](#) with our Python 3 overview.

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)

## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 4: Extraemos el contenido que hemos descargado y lo instalamos:

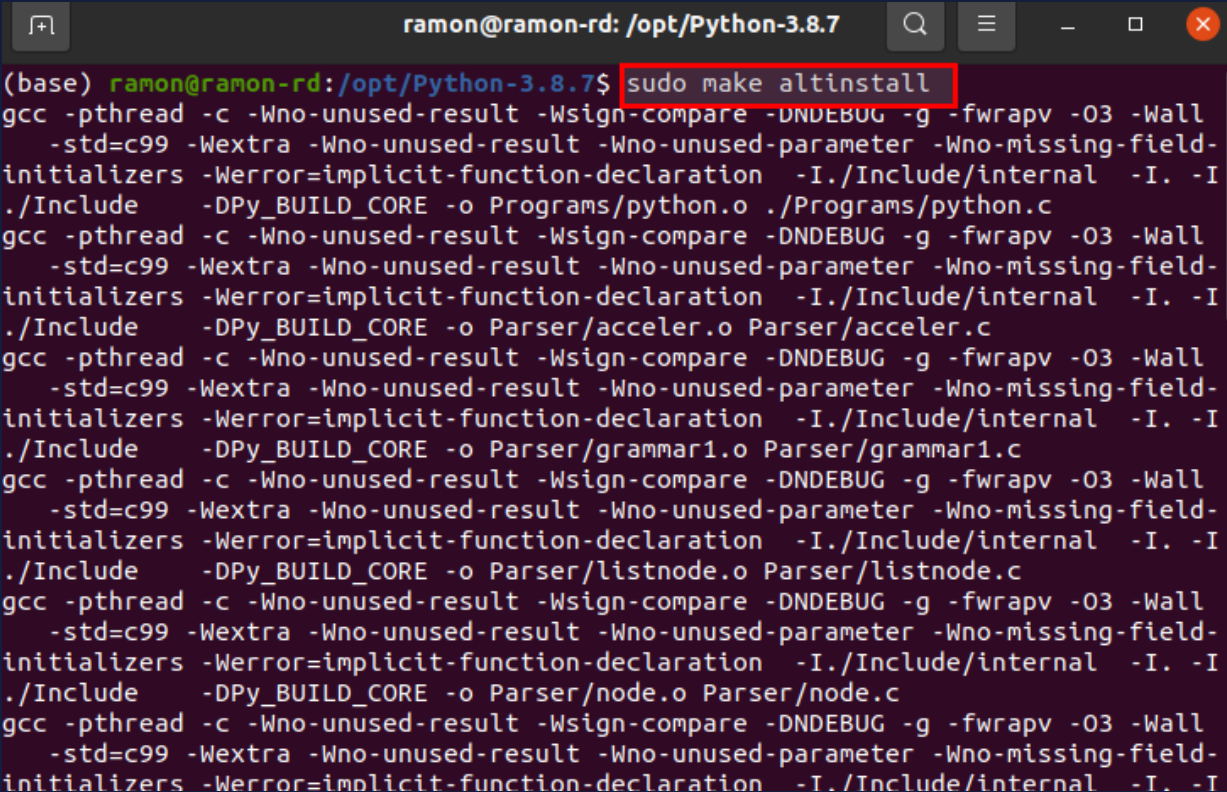
```
sudo tar xzf Python-3.8.7.tgz
cd Python-3.8.7
```

A terminal window titled 'ramon@ramon-rd: /opt/Python-3.8.7' showing the execution of 'sudo ./configure --enable-optimizations'. The terminal output lists various system checks for build system type, host system type, python3.8, universal SDK, archs, MACHDEP, gcc, C compiler, and various options. It concludes with a message about building C++ extension modules.

```
ramon@ramon-rd: /opt/Python-3.8.7
(base) ramon@ramon-rd:/opt/Python-3.8.7$ sudo ./configure --enable-optimizations
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for python3.8... python3.8
checking for --enable-universalsdk... no
checking for --with-universal-archs... no
checking MACHDEP... "linux"
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for a sed that does not truncate output... /usr/bin/sed
checking for --with-cxx-main=<compiler>... no
checking for g++... no
configure:
By default, distutils will build C++ extension modules with "g++".
```

## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Paso 4: Extraemos el contenido que hemos descargado y lo instalamos:



```
ramon@ramon-rd: /opt/Python-3.8.7
(base) ramon@ramon-rd:/opt/Python-3.8.7$ sudo make altinstall
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Programs/python.o ./Programs/python.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/acceler.o Parser/acceler.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/grammar1.o Parser/grammar1.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/listnode.o Parser/listnode.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/node.o Parser/node.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
```

## Preparando el entorno de desarrollo. Instalación de Python en Ubuntu 16.04

Para comprobar que la instalación se ha realizado correctamente ejecutamos el siguiente comando:

```
python3.8 -V
```

NOTA: si ejecutáis en vuestra terminal el comando `python -V` no obtendremos como salida la versión que acabamos de instalar. Para solucionarlo, ejecutaremos las siguientes instrucciones para establecer en nuestro sistema por defecto la versión 3.8.7 de Python:

```
echo "alias python=python3.8" >> ~/.bashrc  
source ~/.bashrc
```

## Gestor de paquetes Pip en Linux

El gestor de paquetes pip nos permitirá instalar, actualizar y eliminar paquetes que utilizaremos en nuestros programas, como por ejemplo numpy, pandas o seaborn.

```
python -m pip install paquete // Instalar la última versión
```

```
python -m pip uninstall [opciones] <paquete> ...
```

**<https://pip.pypa.io/en/stable/reference/pip/>**

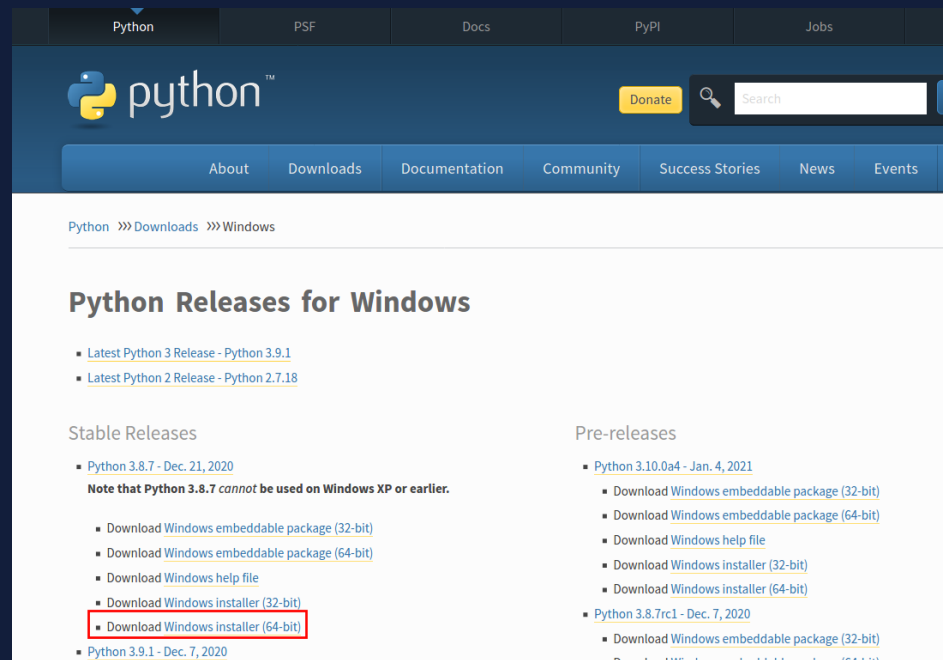
## **Preparando el entorno de desarrollo. Instalación de Python en Windows 10**

A diferencia de los sistemas UNIX, Windows no incluye un sistema que soporte la instalación de Python.

Sin embargo, para poder utilizar Python en Windows, los desarrolladores de Cpython han compilado un conjunto de instaladores (paquetes MSI) para poder llevar a cabo la instalación.

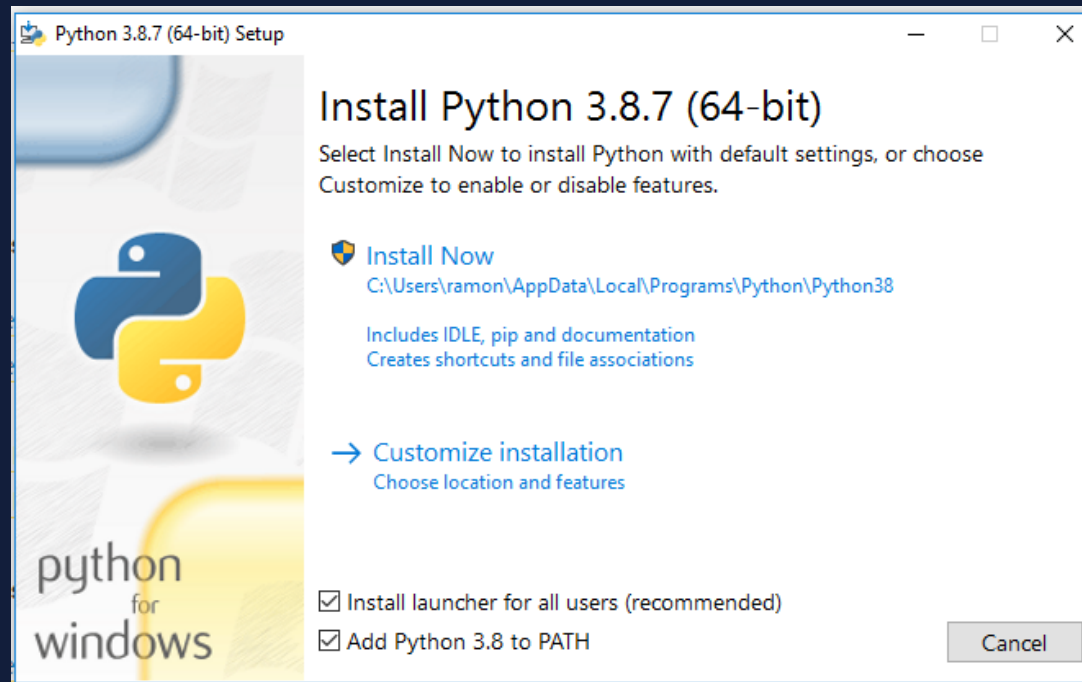
# Preparando el entorno de desarrollo. Instalación de Python en Windows 10

Paso 1: Accedemos a la página oficial de Python, a la sección de descargas para Windows, y descargamos la versión 3.8.7. En mi caso, mi máquina es de 64 bits.



## Preparando el entorno de desarrollo. Instalación de Python en Windows 10

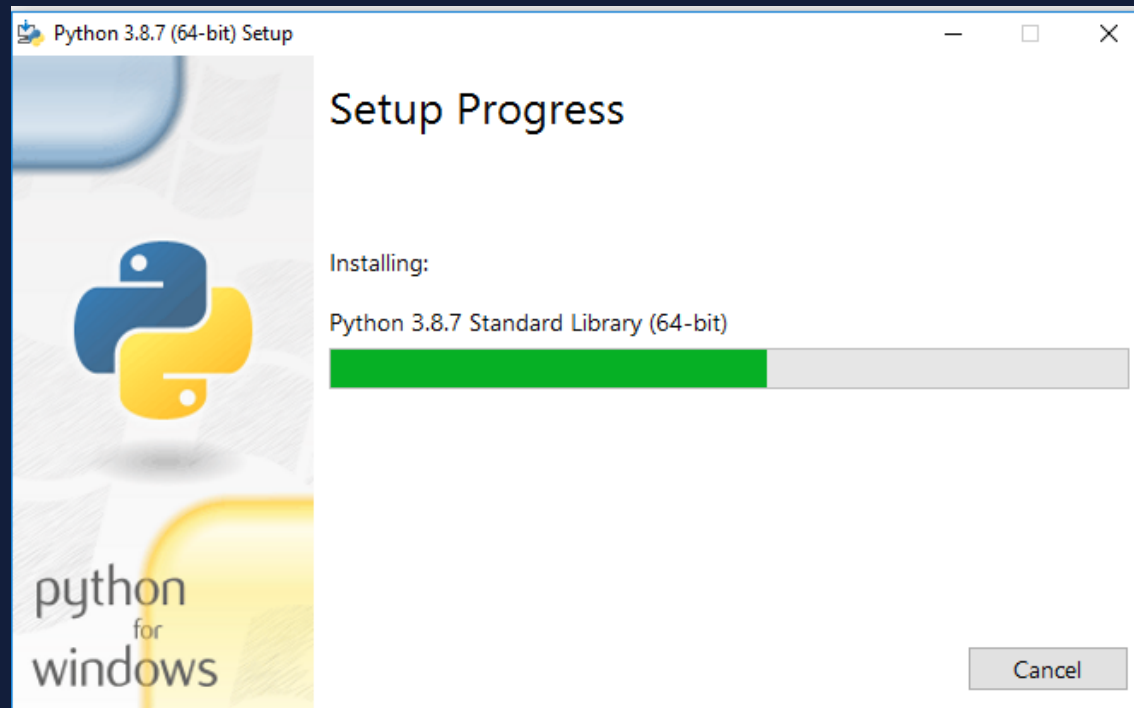
Paso 2: Ejecutamos el fichero descargado y comenzamos con la instalación.





## Preparando el entorno de desarrollo. Instalación de Python en Windows 10

Paso 2: Ejecutamos el fichero descargado y comenzamos con la instalación.



## Preparando el entorno de desarrollo. Instalación de Python en Windows 10

Paso 2: Ejecutamos el fichero descargado y comenzamos con la instalación.



## **Preparando el entorno de desarrollo. Instalación de Python en Windows 10**

Paso 3: Para comprobar que la instalación ha sido realizada con éxito, escribimos en una terminal la orden: `python --version`

## Preparando el entorno de desarrollo. Instalación de Python en Windows 10

Al igual que sucede en el sistema operativo linux, Windows también dispone del gestor de paquetes pip. Este comando nos permitirá, entre otras, instalar, actualizar y eliminar paquetes que serán utilizados en nuestras implementaciones.

```
py -m pip install SomePackage          # última versión
```

```
python -m pip uninstall [opciones] <package> ...
```

<https://pip.pypa.io/en/stable/reference/pip/>

## CONCLUSIONES

1

Python se encuentra entre los mejores lenguajes de programación utilizados actualmente.

2

Su posición estratégica lo ha convertido en una herramienta esencial para cualquier experto.

3

La importancia de saber instalar Python en cualquier sistema operativo y saber realizar una correcta gestión de paquetes.



MUCHAS GRACIAS POR SU ATENCIÓN



[ramonruedadelgado@gmail.com](mailto:ramonruedadelgado@gmail.com)



Ramón Rueda Delgado  
<https://www.linkedin.com/in/ramon-rueda/>



[twitter.com/eiposgrados](https://twitter.com/eiposgrados)



[facebook.com/eiposgrados](https://facebook.com/eiposgrados)



[instagram.com/eiposgrados](https://instagram.com/eiposgrados)